

**REMARKS**

Applicant respectfully requests reconsideration and allowance of the subject application. Claims 18-19 are canceled. Claims 1, 16, 17, 24, 27, and 33 are amended. Accordingly, claims 1-17 and 20-37 remain pending.

Applicant thanks the Examiner for the detailed analysis presented in the preceding Office Actions.

**CLAIM REJECTIONS UNDER 35 U.S.C. § 103**

Claims 1-26, 33, and 37 were rejected under 35 U.S.C. § 103 as being as unpatentable over U.S. Patent No. 6,629,128 B1 to Glass (hereinafter "Glass") in view of U.S. Patent No. 6,560,591 B1 to Memmott et al. (hereinafter "Memmott"). Applicants have carefully considered the reasoning expressed in the preceding Office Actions. Applicants respectfully traverse the rejection, and submit that the claims, as amended, are in condition for allowance.

The subject application is directed to challenges faced in managing systems and devices in an enterprise environment. As computer systems and networks continue to increase in size and complexity, so too does the challenge of managing them. A significant tool that assists network developers and administrators in managing computers across an enterprise is Windows® Management Instrumentation (WMI). WMI enables the remote management of Windows-based systems and applications by exposing management information through an object-oriented structure defined using WMI schemas. WMI schemas are an implementation of the Common Information Model (CIM) as defined by the Desktop Management Task Force (DMTF).

1 WMI supports the management of systems and devices by exposing  
2 management information across an enterprise, such as hardware settings,  
3 performance information, driver configurations, BIOS information, application  
4 settings, event log information, and so on, and by providing a mechanism to query  
5 for information and configure settings on machines across the enterprise. WMI  
6 provides access to management information on a single network machine, or a  
7 large number of machines all at once. For example, without WMI, an  
8 administrator wanting to enumerate descriptions for various groups of objects on a  
9 machine must locate and learn different application programming interfaces  
10 (APIs) that describe the specific methods for communicating with each group of  
11 objects. However, WMI eliminates the need to learn the specifics of every API set  
12 provided by Windows, through gathering information from a diverse set of APIs  
13 and presenting this information in a simple, industry-standard management object  
14 model.

15 Therefore, many comprehensive and well-documented managed resources  
16 are available to those developers and administrators capable of utilizing the  
17 benefits of WMI. However, WMI is generally designed for use by developers or  
18 administrators who are at least moderately proficient at programming in C/C++,  
19 Microsoft Visual Basic<sup>®</sup>, and scripts.

20 Writing such scripts, however, may be beyond the ability of many  
21 administrators, and the discovery of basic system information may therefore be  
22 difficult without the assistance of a more experienced programmer. In addition,  
23 much of the power of WMI is realized through developers writing management  
24 applications that monitor, configure and control the management information  
25 made available through WMI. Therefore, the benefits of WMI are often difficult

1 to attain for the common administrator who does not have the proper programming  
2 background, but who still has a need to manage system components/objects.

3 Independent claim 1 has been amended, as have the other independent  
4 claims. claim 1, as amended, is reproduced below:

5  
6 1. A command line utility embodied in one or more  
7 computer-readable media, the command line utility comprising:

8 a command schema including one or more commands  
9 enabling at least one of retrieval of management information from  
10 and initiation of a management service available through an object  
11 model target schema recognized by at least one target station  
12 accessible over a network;

13 an interactive user interface configured to receive the  
14 one or more commands in the command schema from a user and  
15 communicate a response of the at least one target station to the user;  
16 and

17 an object model command schema to define a mapping  
18 between the one or more commands in the command schema and the  
19 an object model target schema and interpret the one or more  
20 commands from generated by the command schema to cause one of  
21 the retrieval of management information from and the initiation of  
22 the management service on the at least one target and configured to  
23 operate against the target schema through the command line utility.  
24  
25

1 Applicants respectfully call attention to the second paragraph, reciting the  
2 interactive user interface.

3 Glass describes a system that facilitates communications between two  
4 different programs by creating proxies allowing a client application to  
5 communicate with a server-based object:

6 According to an embodiment of the present invention, a system for  
7 distributed processing in a computer network is provided that  
8 includes, a client side object request broker executing on a client  
9 computer and a server-side object request broker executing on a  
10 server computer. The server computer is connected to the client  
11 computer through a network. *A remote proxy generator dynamically*  
12 *generates remote proxy classes for client-side communications*  
13 *support for communications between a client application and a*  
14 *server object.* The remote proxy generator resides in the server-side  
15 object request broker and instantiates the remote proxy class to  
16 create a remote proxy object. A client-side type generator generates  
17 a client side type object for a class of the server object. *The client-*  
18 *side type object provides access to methods of the server object. A*  
19 *client-side function generator generates one or more client-side*  
20 *function objects for providing a connection to one or more*  
21 *methods of the server object.*

22 (Glass, Column 3, Line 66, through Column 2, Line 12; emphasis added).

23 In this context, it is clear that Glass's description of an "interface" is an  
24 interface between programs, not a user interface. The inter-object interface  
25 contemplated by Glass is defined in the first paragraph of the background of the  
invention:

Classes may also be characterized by their *interface which defines*  
*the elements necessary for proper communication between objects.*

(Glass, Column 1, Lines 29-31; emphasis added). Furthermore, while there a few  
mentions of users in the background of the invention of Glass (See Glass, Column  
1, Line 46; Column 3, Lines 54-59), the word "user" is never mentioned in the  
detailed description of the invention of Glass. Moreover, the phrase "user

1 interface" is never mentioned at all. Because Glass does not describe a user  
2 interface, applicants respectfully submit that Glass neither teaches nor suggest the  
3 elements recited in claim 1.

4 Applicants acknowledge the Office Action's recitation of "a command line  
5 predevelopment utility" (*Glass*, Column 19, lines 10-14). Respectfully, however,  
6 the "command line predevelopment utility," when considered in context, is not a  
7 user interface at all. The "command line predevelopment utility" is an alternative  
8 inter-object communication mechanism used by Glass to facilitate communication  
9 between software objects that do not include a suitable inter-object interface:

10 Referring to FIG. 11, *an interface generator 250 is*  
11 *illustrated for use in remote enabling classes without interfaces.* A  
12 typical remote proxy 154 resides in client system 102 and  
13 communicates through network 106 with server object 110 using  
14 server object interface 111. Existing class files on server system 104  
15 may need to be used remotely by client application 108 on client  
16 system 102. Before the existing class file may be used remotely, it  
17 should have an interface in order to comply with the communication  
18 standards of typical ORBs. Interface generator 250 generates an  
19 interface 254 for a class file 252. Interfaces provide for inheritance  
20 from multiple sources and ease of method invocation. *Without*  
21 *interfaces, a complex procedure using reflection is used to invoke*  
22 *methods directly on objects.*

23 *In one embodiment, interface generator 250 is a command*  
24 *line predevelopment utility used to generate interfaces for classes*  
25 *on server system 104 that will be used remotely in distributed*  
*computing system 100. In that embodiment, the software developer*  
*knows that certain class files 252 will be used remotely. The*  
*software developer provides interface generator 250 with a list of*  
*class files 252 for which interfaces 254 are to be generated.*

21 (*Glass*, Column 18, Line 63, through Column 19, Line 17; emphasis added). This  
22 appears to be the only mention of a "command line" included within Glass. It is  
23 clear that the "command line" as used in Glass is part of "predevelopment utility"  
24 enabling a "software developer" to identify objects for which an inter-object  
25 interface will be needed; respectfully, it is just as clear that the "command line

1 predevelopment utility" is not a "user interface," as recited in claim 1. The only  
2 interface taught or suggested by Glass is an interface between program objects.  
3 Thus, Glass fails to teach or suggest the subject matter of claim 1.

4 Applicants note that the other independent claims also have been amended  
5 to recite or clarify inclusion of a user interface. A user interface is recited, in  
6 pertinent part, in each of the independent claims as recited below:

7 • Claim 16, as amended:

8 an interactive user interface configured to receive one or more  
9 commands in the command schema from a user and communicate a  
10 response of the at least one target station to the user; and

11 • Claim 17, as amended:

12 an interactive interface utility configured to receive a user  
13 command to facilitate implementation of individual commands  
14 within the set of commands, wherein the interactive interface utility  
15 includes at least one of:

16 a command line interface utility configured to receive textual  
17 entry of the user command; and

18 a graphical user interface utility.

19 • Claim 24, as amended:

20 an interactive interface configured to receive the set of  
21 commands from a user, wherein the interactive interface includes at  
22 least one of:

23 a command line interface utility configured to receive textual  
24 entry of the user command; and

25 a graphical user interface utility.

- Claim 33, as amended:

providing an interactive user interface configured to receive commands from a user;

Applicants submit that each of claims 16, 17, 24, and 33 recite at least one element that is neither taught nor disclosed by Glass. Thus, applicants request that the rejections under 35 U.S.C. § 103 be withdrawn from claims 1, 16, 17, 24, and 33.

For these same reasons, applicants request that the rejections under 35 U.S.C. § 103 be withdrawn from claims 2-15, 20-23, 25-26, and 34-37. Because dependent claims 2-15, 20-23, 25-26, and 34-37 apply additional limitations to the claims from which they depend, these claims are patentable for at least the same reasons as the claims from which each depends, as previously described. Thus, applicants respectfully request that the rejection under 35 U.S.C. § 103 be withdrawn with regard to 2-15, 20-23, 25-26, and 34-37.

In addition, although applicants do not concede that one of ordinary skill in the art at the time the invention was made would supplement Glass with Memmott, applicants submit that Memmott fails to make up for the shortcomings of Glass. Independent claim 1, in pertinent part, recites mapping of commands to object model target schema:

[A]n object model command schema to define a mapping between the one or more commands in the command schema and the object model target schema and interpret the one or more commands from the command schema and received via the user interface to cause one of the retrieval of management information from and the initiation of the management service on the at least one target.

Respectfully, Memmott neither teaches nor discloses what is recited in claim 1.

Memmott, in pertinent part, "free[s] a requesting entity from the burden of having to select a particular provider" (*Memmott*, Column 3, Lines 7 and 8). More

1 particularly, Memmott discloses a "data resolver" that is used to retrieve  
2 information for a "data requester," regardless of which "data provider" maintains  
3 the requested data (*Memmott*, Column 3, Lines 7-67). Accordingly, Memmott  
4 provides access to data without the data requester having to know where the data  
5 resides, freeing the data requester from having to manage the information or the  
6 services that provide it.

7 On the other hand, while Memmott frees the data requester from having to  
8 know what data provider stores the desired information, Memmott requires the  
9 data requester to formulate a request for data in the format recognized by the data  
10 provider:

11 Possible formats for the query received from data requestor  
12 110 include object-oriented formats such as Managed Object Format  
13 (MOF) and syntaxes such as Extensible Markup Language (XML).  
14 For example, the query may conform to at least one among the  
15 distributed management schemes referenced above (SNMP, CMIP,  
16 DMI, CIM) or to a similar scheme such as Windows Management  
17 Interface (WMI, Microsoft Corp., Redmond, Wash.). Included in the  
18 query is a query characteristic that identifies the information  
19 requested and/or the subject matter of the query. For example, a  
20 query relating to a DVD (Digital Versatile Disk) drive may include  
21 an object class (e.g. MediaAccessDevice), a subclass (e.g.  
22 DVDDrive), and an indication of the particular drive property about  
23 which information is desired

24 (*Memmott*, Column 3, Lines 27-42). Thus, while Memmott provides the data  
25 requester some freedom in identifying the data provider, the data requester must  
26 request data in the manner dictated by the data provider. Accordingly, because  
27 Memmott is directed to provider transparency, applicants submit that Memmott  
28 fails to redress shortcomings of Glass. Thus, applicants submit that the claims are  
29 allowable over Glass in view of Memmott.



1 Claims 27-32 were rejected under 35 U.S.C. § 103 as being as unpatentable  
2 over Memmott in view of Glass and in further view of Steve, "Network and  
3 System Management with XML" (hereinafter "Nash").

4 Independent claim 27 is amended as reproduced below:

5  
6 27. (Currently amended) A method comprising:  
7 receiving a command from a user through an  
8 interactive command line interface;  
9 fetching an alias for the command;  
10 interpreting the command based on the alias and the  
11 current operating environment of the command line interface;  
12 executing the command as one or more WMI API calls  
13 against a targeted namespace representing at least one target system  
14 that is accessible via a network;  
15 receiving WMI data in XML form;  
16 applying an XSL style sheet format the WMI data; and  
17 presenting the WMI data through the command line  
18 interface.

19  
20 Applicants respectfully call attention to the second and third paragraphs, reciting  
21 fetching an alias and interpreting the command based on the alias.

22 As previously described, Memmott requires the data requester to formulate  
23 a request for data in the format recognized by the data provider. Although  
24 Memmott may provide location transparency, it provides no command  
25 interpretation. Moreover, although applicants do not concede that one of ordinary

1 skill in the art at the time the invention was made would supplement Memmott  
2 with Glass, for the reasons described previously, applicants submit that Glass fails  
3 to make up for the shortcomings of Memmott. Further, applicants submit that  
4 Nash fails to make up for the shortcomings of Memmott or Glass, either alone or,  
5 for sake of argument, combined to teach or suggest the subject matter of claim 27.  
6 Thus, applicants request that the rejection under 35 U.S.C. § 103 be withdrawn  
7 from claim 27.

8 For these same reasons, applicants request that the rejections under 35  
9 U.S.C. § 103 be withdrawn from claims 28-32. Because dependent claims 28-32  
10 apply additional limitations to the claim from which they depend, these claims are  
11 patentable for at least the same reasons as the claims from which each depends, as  
12 previously described. Thus, applicants respectfully request that the rejection under  
13 35 U.S.C. § 103 be withdrawn with regard to 28-32.

CONCLUSION

Claims 1-17 and 20-37 are in condition for allowance. Applicant respectfully requests reconsideration and prompt allowance of the subject application. If any issue remains unresolved that would prevent allowance of this case, the Examiner is requested to urgently contact the undersigned attorney to resolve the issue.

Respectfully Submitted,

Date: Aug. 3, 2005

By: 

Frank J. Bozzo  
Lee & Hayes, PLLC  
Reg. No. 36,756  
(206) 315-4001